Remarks

In the Office Action, the Examiner noted that claims 1-31, 33 and 34 are pending in the application, and that claims 1-31, 33 and 34 are rejected. By this amendment, claims 1, 10, 21, 22, and 30 have been amended, and new claims 35-44 have been added. Thus, claims 1-31 and 33-44 are pending in the application.

Applicant hereby requests further examination and reconsideration of the application, in view of the foregoing amendments.

In the Claims

Rejection Under 35 USC 103(a)

The Examiner rejected claims 1-6 under 35 U.S.C. § 103(a), as being taught by *Popescu et al.*, U.S. Patent No. 5,487,156 (hereinafter *Popescu*) in view of Priess's "Direct vs. Indirect Containment" (hereinafter *Priess*). Applicant respectfully traverses.

Claims 1-7

With respect to claim 1, the Examiner asserts that it would have been obvious to a person of ordinary skill in the art at the time the invention was made to incorporate the direct storage of store instruction results of *Priess* in the device of *Popescu* to simplify the cache implementation and increase the speed of the processor. Applicant respectfully disagrees and asserts that the Examiner has failed to establish a *prima facie* case of obviousness because: 1) the references cited by the Examiner do not teach the claim limitation of a result forwarding cache (RFC) that stores store instruction results; and 2) the references do not provide a motivation to combine them in the manner recited by the amended claim; in particular, they do not suggest avoiding stalling a load instruction until storehit data is updated in the processor's data cache, which is the problem solved by the amended claimed invention. Both of these reasons are discussed at length below; however, first a discussion of *Priess* is helpful.

The *Priess* reference cited by the Examiner is a web page of a web book entitled: "Data Structures and Algorithms with Object-Oriented Design Patterns in C++", which is displayed at the top of the web page. *Priess* teaches putting a C++ software language object into a container (also a C++ software language object) by either making a copy of the object in the container (direct containment) or keeping a pointer to the object in the container (indirect containment). *Priess* also teaches that a disadvantage of direct containment is that an object cannot be contained in more than one container at a time.

A. The References Do Not Teach a RFC that Stores Store Instruction Results

The Examiner acknowledges in the instant Office Action that *Popescu* does not teach a RFC that stores a plurality of store instruction results. Furthermore, Applicant respectfully asserts that *Priess* does not teach a RFC. A RFC, as recited by amended claim 1, is a part of an apparatus in a pipelined microprocessor that stores at least one non-store instruction result destined for a user-visible register of the microprocessor and stores a plurality of store instruction results destined for a data cache of the microprocessor. In contrast, *Priess* teaches putting a C++ software language object into a container (also a C++ software language object) by either making a copy of the object in the container or keeping a pointer to the object in the container. *Priess* does not even teach a microprocessor, much less a pipelined microprocessor, much less an apparatus within a pipelined microprocessor for forwarding store data to a load instruction, much less an RFC, much less a RFC that stores store data. Consequently, Applicant respectfully asserts that *Popescu* in view of *Priess* does not obviate amended claim 1 because neither of the references teaches the recited RFC claim element. Therefore, Applicant respectfully asserts the Examiner has failed to make a *prima facie* case of obviousness, and requests the Examiner withdraw the rejection.

B. The References Do Not Provide a Motivation to Combine the References in the Manner Claimed

To make a *prima facie* case of obviousness, the Examiner must show a teaching, suggestion, or motivation provided by the references to combine the references; in particular, the Examiner may not in hindsight obtain the motivation to combine the

references from the Applicant's disclosure.[1] The Examiner stated two motivations to combine the *Popescu* and *Priess* references: 1) to increase the speed of the processor, and 2) to simplify the cache implementation. Applicant respectfully asserts the Examiner has failed to show that the references provide a motivation to combine them in the manner claimed.

Applicant explains in his Background section that the problem with prior processors is that when a storehit occurs – i.e., a load instruction specifies a load address that matches a store address of a previous store instruction whose store data is still in the microprocessor's pipeline and yet to be updated in the microprocessor's data cache – the load instruction is stalled until the store data is updated in the processor's data cache for availability to the load instruction.[2] See page 7, line 6 to page 8, line 2. The invention recited in amended claim 1 solves this problem by providing a RFC from which storehit data may be forwarded to a load instruction, without having to stall the load instruction until the storehit data is updated to the processor's data cache. However, none of the references cited by the Examiner provide a teaching, suggestion, or motivation to solve this problem.

First, *Popescu*'s processor operates like the processors described in Applicant's Background section, i.e., it stops the load instruction from accessing memory until the store data is updated to memory. See Col. 8, lines 41-44, 48-50. Second, as discussed above, the *Priess* reference is not even in the field of pipelined microprocessors, much less of stalling a load instruction in a microprocessor or forwarding storehit data within a microprocessor pipeline, and therefore does not provide a teaching, suggestion, or

---

[1] "'Obviousness cannot be established by combining the teachings of the prior art to produce the claimed invention, absent some teaching or suggestion supporting the combination. Under section 103, teachings of references can be combined *only* if there is some suggestion or incentive to do so.'" *In re Fritch*, 972 F.2d 1260, 1266 (Fed. Cir. 1992) (quoting *ACS Hosp. Systems, Inc. v. Montefiore Hosp.*, 732 F.2d 1572, 1577, 221 USPQ 929, 933 (Fed. Cir. 1984)). "The mere fact that the prior art may be modified in the manner suggested by the Examiner does not make the modification obvious unless the prior art suggested the desirability of the modification. ... It is impermissible to use the claimed invention as an instruction manual or 'template' to piece together the teachings of the prior art so that the claimed invention is rendered obvious. This court has previously stated that '[o]ne cannot use hindsight reconstruction to pick and choose among isolated disclosures in the prior art to deprecate the claimed invention.'" *Id.* (quoting *In re Fine*, 837 F.2d 1071, 1075, 5 USPQ 2d 1596, 1600 (Fed. Cir. 1988)).

[2] The alternative discussed in Applicant's Background section is a result distribution bus, which becomes prohibitive as the number of pipeline stages increases.

motivation to solve the problem identified by the Applicant. In fact, *Priess* teaches away from the notion of caching altogether, since it teaches that a disadvantage of direct containment is that an object cannot be contained in more than one container at a time; whereas, a basic premise and purpose of caching is that the cached data is stored in two hardware structures at the same time, namely the larger main memory and the smaller cache memory, so that the cache memory can be accessed more quickly. It is not surprising that *Priess* teaches away from the notion of caching since it is in the non-analogous field of software languages rather than pipelined microprocessor hardware. Third, the FOLDOC cache definition reference merely teaches generally the notion of including a data cache in a CPU in order to reduce access times over main memory, but teaches nothing about pipelined microprocessors, much less of stalling a load instruction in a microprocessor or forwarding storehit data within a microprocessor pipeline, and therefore does not provide a teaching, suggestion, or motivation to solve the problem identified by the Applicant. Consequently, since none of the references provides a teaching, suggestion, or motivation to solve the problem identified by the Applicant, Applicant respectfully suggests that the Examiner is in hindsight using Applicant's disclosure as the motivation to piece together the references to create the claimed invention, which is improper. The teaching, suggestion, or motivation must be provided by the references. Applicant respectfully asserts that the Examiner has failed to make a *prima facie* case of obviousness, and the rejection is improper. Applicant respectfully requests the Examiner withdraw the rejection.

As a further note, Applicant respectfully suggests the Examiner has misunderstood the operation of *Popescu*'s DRF. The Examiner appears to be indicating that the DRF functions as a cache of store addresses, or pointers, so that the pointers can be used to access main memory to obtain the load data. This is incorrect. The load address is already generated before it is compared with the store addresses in the DRF; i.e., the load address is not obtained from the DRF. See Col. 8, lines 31-32. Rather, the purpose of storing the store addresses in the DRF is to be able to determine whether the load instruction must be stalled until the store data whose address matches the load instruction is sent to memory, as discussed above. See Col. 8, lines 48-50.

Applicant respectfully asserts *Popescu* in view of *Priess* does not obviate dependent claims 2-6 because they depend from independent claim 1, which is not obviated by *Popescu* in view of *Priess* for the reasons discussed above.

The Examiner rejected claim 7 under 35 U.S.C. § 103 as being unpatentable over *Popescu* in view of *Priess* and in further view of *In re Rose*, 220 F.2d 459 (CCPA 1955). Applicant respectfully traverses the Examiner's rejections. Applicant respectfully asserts *Popescu* in view of *Priess* and in further view of *In re Rose* does not obviate dependent claim 7 because it depends from independent claim 1, which is not obviated by *Popescu* for the reasons discussed above.

Claims 8-9

The Examiner rejected claims 8 and 9 under 35 U.S.C. § 103 as being unpatentable over *Popescu* in view of *Priess* and in further view of *Abramson et al*, U.S. Patent No. 5,606,670 (hereinafter *Abramson*). Applicant respectfully traverses the Examiner's rejections. Applicant respectfully asserts *Popescu* in view of *Priess* and in further view of *Abramson* does not obviate dependent claims 8 and 9 because they depend from independent claim 1, which is not obviated by *Popescu* in view of *Priess* for the reasons discussed above.

Claims 10-20 and 26-29

The Examiner rejected claims 10-20 and 26-29 under 35 U.S.C. § 103 as being unpatentable over *Abramson* in view of *Popescu*. Applicant respectfully traverses the Examiner's rejections.

With respect to claim 10, the Examiner asserts that *Popescu* teaches an RFC configured to forward a first plurality of store instruction results. In the Response to Arguments, the Examiner stated that, unlike claim 1, claim 10 did not recite the limitation that the RFC stores the plurality of store instruction results. Applicant has amended claim 10 to recite the limitation. Therefore, Applicant respectfully asserts that *Popescu* does not teach an RFC configured to store and forward a first plurality of store instruction results as recited in amended claim 1, and for reasons similar to those discussed above with respect to

claim 1. Further, the Examiner asserts that a person of ordinary skill in the art would have recognized that the RFC reduces stalls in a pipeline. The motivation cited by the Examiner is the problem stated in Applicant's background section, which the references cited by the Examiner do not provide a teaching, suggestion, or motivation to solve, as discussed above; therefore, Applicant respectfully suggests the Examiner is obtaining the motive to reduce pipeline stalls from Applicant's disclosure, which is improper. Therefore, Applicant respectfully asserts the Examiner has failed to make a *prima facie* case of obviousness with respect to claim 10 by citing *Abramson* in view of *Popescu*, and Applicant requests the Examiner withdraw the rejection.

Applicant respectfully asserts *Abramson* in view of *Popescu* does not obviate dependent claims 11-17 because they depend from independent claim 10, which is not obviated by *Abramson* in view of *Popescu* for the reasons discussed above.

With respect to claim 18, the Examiner asserts that *Abramson* teaches first and second comparison logic for comparing the load address with first and second plurality of store addresses of first and second store instruction data in a plurality of pipeline stages and in a plurality of store buffers, respectively, citing the same text in *Abramson* for both the first and second comparison logic. Applicant can only find comparators that compare a load address with store addresses of store instruction data in store buffers, but cannot find comparators that compare a load address with store addresses of store instruction data anywhere else in *Abramson*'s processor. Therefore, Applicant respectfully asserts it is improper for the Examiner to recite *Abramson* as teaching both the first and second comparison logic recited in claim 18 since the claimed first and second comparators compare store addresses of distinctly characterized data; whereas *Abramson* teaches comparators for comparing only similarly characterized data, namely all in store buffers.

Further with respect to claim 18, the Examiner asserts that *Popescu* teaches a plurality of stages of a pipeline that does not include store buffers, citing text of *Popescu* that teaches the DRF. Furthermore, although the Examiner does not explicitly state it, the Examiner must be implying that *Popescu*'s processor forwards store instruction data from the DRF to the load instruction, since the claim recites forwarding store instruction data to the load

instruction from the stages comprising non-store buffers, which the Examiner is asserting as *Popescu*'s DRF, and *Abramson* does not teach forwarding store instruction data from a pipeline stage that includes non-store buffers. However, as discussed above with respect to claim 1, *Popescu* does not teach forwarding store instruction data stored in the DRF. Therefore, Applicant respectfully asserts the Examiner has failed to make a *prima facie* case of obviousness with respect to claim 18, and respectfully requests the Examiner withdraw the rejection.

Applicant further notes that the Examiner in the Response to Arguments cites *In re Keller* and *In re Merck & Co.* throughout for the proposition that one cannot show non-obviousness by attacking the references individually where the rejections are based on combinations of references. However, Applicant notes that the Examiner is citing from § 2145 of the MPEP, which addresses an Examiner's consideration of an Applicant's rebuttal arguments *after* a *prima facie* case of obviousness has already been made, which Applicant is not attempting to do here. Furthermore, the cited statements from both *In re Keller* and *In re Merck & Co.* are made in the context of rebutting obviousness *after* a *prima facie* case of obviousness has already been made. However, Applicant's arguments are being made to show that the Examiner has failed to make a *prima facie* case of obviousness, namely to show the Examiner is relying on a reference to teach a claim limitation that the reference does not teach, or that the Examiner has failed to distinctly point out where a reference teaches a limitation, not to rebut a showing of a *prima facie* case of obviousness. Thus, it is proper for the Applicant to attack a reference individually to argue that the Examiner has failed to make a *prima facie* case of obviousness rather than to rebut a *prima facie* case of obviousness.

Applicant respectfully asserts *Abramson* in view of *Popescu* does not obviate dependent claims 19-20 because they depend from independent claim 18, which is not obviated by *Abramson* in view of *Popescu* for the reasons discussed above.

With respect to claim 26, the Examiner asserts that *Popescu* in view of *Priess* teaches storing at least one store instruction result and at least one non-store instruction result into a result forwarding cache of a microprocessor. Applicant respectfully asserts that

Popescu in view of *Priess* does not teach storing at least one store instruction result and at least one non-store instruction result into a result forwarding cache of a microprocessor for the reasons stated above with respect to claims 1 and 10.

Furthermore, with respect to claim 26, the Examiner asserts that *Abramson* teaches determining whether storehit data is present in a result forwarding cache. Applicant respectfully asserts that *Abramson* does not teach a RFC. Applicant can only find that *Abramson* teaches store data buffers of a memory order buffer (MOB) for storing store data, as described in the text of *Abramson* cited by the Examiner, not a RFC for storing both a store instruction result and a non-store instruction result.

Still further, with respect to claim 26, the Examiner asserts that *Abramson* teaches selectively forwarding the data from the result forwarding cache. Applicant respectfully asserts that *Abramson* does not teach forwarding the data from a result forwarding cache because, as just discussed, *Abramson* does not teach a RFC.

Thus, for the reasons stated above, Applicant respectfully asserts the Examiner has failed to make a *prima facie* case of obviousness with respect to claim 26 by citing *Abramson* in view of *Popescu* further in view of *Priess*, and Applicant respectfully requests the Examiner withdraw the rejection.

Applicant respectfully asserts *Abramson* in view of *Popescu* does not obviate dependent claims 27-29 because they depend from independent claim 26, which is not obviated by *Abramson* in view of *Popescu* for the reasons discussed above.

Claims 21-25, 30-31, and 33-34

The Examiner rejected claims 21-25, 30-31, and 33-34 under 35 U.S.C. § 103 as being unpatentable over *Abramson* in view of *Popescu* and in further view of Patterson and Hennessy's Computer Architecture: A Quantitative Approach, Second Edition © 1996 (hereinafter *Hennessy*). Applicant respectfully traverses the Examiner's rejections.

With respect to claim 21, the claim has been amended to clearly and distinctly claim the subject matter Applicant regards as his invention, and Applicant asserts the references

cited by the Examiner neither anticipate individually nor obviate in combination amended claim 21 for reasons similar to those discussed above with respect to claims 1, 10, and 18 and below with respect to new claim 35.

Further with respect to amended claim 21, Applicant notes that although *Abramson* refers to "speculative" execution, what *Abramson* means by execution is the possibility that execution results of speculatively executed instructions may have to be purged because of a branch misprediction (see col. 4, lines 54-55), not because of an incorrect determination of newest storehit data based on a virtual address comparison and subsequent physical address comparison as recited in amended claim 21, as also discussed below with respect to new claim 35.

For the reasons stated above, Applicant respectfully asserts the Examiner has failed to make a *prima facie* case of obviousness with respect to claim 21 by citing *Abramson* in view of *Popescu* further in view of *Hennessy*, and Applicant respectfully requests the Examiner withdraw the rejection.

Applicant respectfully asserts *Abramson* in view of *Popescu* and in further view of *Hennessy* does not obviate dependent claim 22 because it depends from independent claim 21, which is not obviated by *Abramson* in view of *Popescu* and in further view of *Hennessy* for the reasons discussed above.

With respect to claim 23, the Examiner states in the Response to Arguments that *Abramson* teaches speculative forwarding of storehit data. Applicant respectfully asserts that *Abramson* does not teach speculative forwarding of storehit data as recited in claim 23, namely forwarding storehit data and subsequently determining that a load address is within a non-cacheable region and therefore should not have been forwarded. Rather, as discussed above with respect to claim 21, the only teaching Applicant can find in *Abramson* regarding speculative execution is with regard to mispredicted branch instructions, which is patentably distinct from speculatively forwarding storehit data as recited in claim 23.

Further with respect to claim 23, the Examiner states in the Response to Arguments that *Hennessy* was relied upon to merely show that stalling the pipeline "after a hazard has been detected" is common practice. However, this statement mischaracterizes claim 23. Certainly it is common practice to stall a pipeline after a hazard has been detected; however, claim 23 recites stalling the pipeline after storehit data has already been speculatively forwarded *and then the hazard has been detected.* Applicant respectfully asserts that neither *Hennessy* nor *Abramson* nor *Popescu* teach stalling the pipeline after storehit data has already been speculatively forwarded and then a hazard has been detected, as recited in claim 23.

For the reasons stated above, Applicant respectfully asserts the Examiner has failed to make a *prima facie* case of obviousness with respect to claim 23 by citing *Abramson* in view of *Popescu* further in view of *Hennessy*, and Applicant respectfully requests the Examiner withdraw the rejection.

Applicant respectfully asserts *Abramson* in view of *Hennessy* does not obviate dependent claims 24-25 because they depend from independent claim 23, which is not obviated by *Abramson* in view of *Hennessy* for the reasons discussed above.

With respect to amended claim 30, Applicant respectfully asserts, for the reasons discussed above with respect to claim 21, that *Abramson* in view of *Popescu* and in further view of *Hennessy* does not obviate claim 30 as amended.

Applicant respectfully asserts *Abramson* in view of *Popescu* and in further view of *Hennessy* does not obviate dependent claims 31 and 33 because they depend from independent claim 30, which is not obviated by *Abramson* in view of *Popescu* and in further view of *Hennessy* for the reasons discussed above.

With respect to claim 34, Applicant respectfully asserts, for the reasons discussed above with respect to claim 23, that *Abramson* in view of *Popescu* and in further view of *Hennessy* does not obviate claim 34.

New Claims 35-44

With respect to claim 35, in the Office Action the Examiner cites col. 1, lines 36-52 of *Abramson* with respect to claim 21. However, in the cited text *Abramson* teaches comparing physical load and store addresses for the purpose of flushing store buffer data, not for the purpose of speculatively forwarding store instruction results. In the cited text *Abramson* also teaches comparing virtual load and store addresses again for the purpose of flushing store buffer data, not for the purpose of speculatively forwarding store instruction results. The Examiner also cites col. 3, line 53 to col. 4, line 10 with respect to claim 21. Although *Abramson* teaches translating virtual addresses to physical addresses and storing the physical addresses of store instructions in a physical address buffer (col. 6, lines 10-12), Applicant can find no teaching in *Abramson* of using the physical addresses to perform store instruction result forwarding. Rather, *Abramson* teaches comparing linear addresses for the purpose of store forwarding. The only teaching Applicant can find in *Abramson* using the physical addresses is for accessing the instruction and data caches. Col. 5, lines 7-8.

Furthermore with respect to claim 35, although *Abramson* teaches speculatively issuing load instructions (col. 2, lines 43-44) and teaches that some instructions are speculatively executed (col. 4, lines 52-53), *Abramson* does not teach speculatively forwarding store instruction results, in the manner recited in claim 35. Rather, *Abramson*'s meaning of speculative execution is executing an instruction based on a branch instruction prediction; and the branch may have been mispredicted and the results of instructions speculatively executed due to the mispredicted branch must be purged. Col. 4, lines 52-55. Thus, *Abramson* does not teach speculatively forwarding store instruction results based on virtual address comparisons and subsequently determining the forwarding was incorrect based on a physical address comparison indicating newer storehit data present in the microprocessor pipeline that was undetected by the virtual address comparison, as recited in claim 35.

For all of the reasons advanced above, Applicant respectfully submits that claims 1-31, and 33-44 are in condition for allowance. Reconsideration of the rejections is requested, and Allowance of the claims is solicited.

Applicant earnestly requests the Examiner to telephone him at the direct dial number printed below if the Examiner has any questions or suggestions concerning the application or allowance of any claims thereof.

Respectfully submitted,

E. Alan Davis

E. Alan Davis
Huffman Law Group, P.C.
Registration No. 39,954
Customer No. 23669
1832 N. Cascade Ave.
Colorado Springs, CO 80907
512.301.7234
512.233.0850 fax
alan@huffmanlaw.net

Date: 3-18-05

"EXPRESS MAIL" mailing label number _____. Date of Deposit _____. I hereby certify that this paper is being deposited with the U.S. Postal Service Express Mail Post Office to Addressee Service under 37 C.F.R. §1.10 on the date shown above and is addressed to the U.S. Commissioner of Patents and Trademarks, Washington, D.C. 20231.
By: _____